STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **13**, May 2025, pp 2027–2043.
Published online in International Academic Press (www.IAPress.org)

Geometric Feature-Based Machine Learning for Efficient Hand Sign Gesture Recognition

Chraa Mesbahi Soukaina [1,*], Masrour Mohammed [2], Rhazzaf Mohamed [3]

[1]*Laboratory of Computer Science, Signals, Automation and Cognitivism (LISAC)-Department of Computer Science-Faculty of Sciences Dhar El Mahraz, Sidi Mohamed Ben Abdellah University, Fez, Morocco*
[2]*Euromed University of Fes, UEMF, Morocco*
[3]*Mohammadia School of Engineers, Mohammed V University in Rabat, Rabat, Morocco*

Abstract Hand Gesture Recognition (HGR) is emerging as a vital tool in enhancing communication, particularly for individuals who are deaf or hard of hearing. Despite its potential, widespread use of sign language remains constrained by limited understanding among the general public. Previous research has explored various models to bridge this communication gap. However, deploying complex deep learning algorithms on low-power, cost-effective embedded devices presents significant challenges due to constraints on memory and energy resources. In this research, we introduce a new approach by leveraging lightweight machine learning algorithms for real-time hand sign recognition, utilizing novel geometrical features derived from hand landmarks. Our approach optimizes computational efficiency without compromising accuracy, making it suitable for resource-limited devices. The proposed model not only achieves higher accuracy compared to existing methods but also demonstrates that a focus on feature design can outperform more complex deep learning architectures, thereby offering a promising solution for real-time, accessible HGR applications.

Keywords Hand Gesture Recognition, Machine Learning, Geometrical Features, Embedded Devices, Classification

DOI: 10.19139/soic-2310-5070-2306

1. Introduction

Today, about 430 million people worldwide are affected by hearing impairment [1], a number that continues to rise as the population ages. For many of these people, sign language is an essential communication tool, enabling them to express their thoughts, feelings, and emotions through a visual language based on gestures [2]. Sign language is not simply a set of hand movements; it is a rich and complex system of communication that relies on the precise positioning and movement of the hands, as well as facial expressions and body language, to convey meaning [3]. Despite its effectiveness within the deaf community, this visual language is not widely understood by the general population, creating a significant communication barrier when people with hearing impairments interact with those who don't know the language. Currently, bridging this communication gap often requires the use of human interpreters, which can be costly and is not always easy to obtain, limiting the independence and social participation of people who depend on sign language.

Hand Gesture Recognition technology has emerged as an effective solution for overcoming communication barriers. HGR systems work by interpreting the complex hand movements used in sign language and converting them into text or speech that can be understood by those unfamiliar with the language. This technology has the

*Correspondence to: Chraa Mesbahi Soukaina (Email: soukaina.chraamesbahi@usmba.ac.ma). Laboratory of Computer Science, Signals, Automation and Cognitivism (LISAC)-Department of Computer Science-Faculty of Sciences Dhar El Mahraz, Sidi Mohamed Ben Abdellah University, Fez, Morocco.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

potential to greatly improve the lives of hearing-impaired individuals by enabling more fluid and spontaneous communication with the general public, without the need for a human interpreter. However, the challenge lies in developing HGR systems that are not only accurate and reliable, but also efficient enough to run on low-cost, portable devices that are practical for everyday use.

Researchers have investigated a number of methods to improve HGR systems. These have included traditional machine learning techniques [4, 5, 6, 7], which rely on manually crafted features, as well as more recent deep learning methods [8, 9, 10], which automatically learn features from data. Traditional machine learning models have the advantage of being less computationally demanding, but they often require extensive feature engineering, which can be challenging given the variability and complexity of hand gestures. In comparison, deep learning models have shown exceptional accuracy and robustness in hand gesture recognition, yet their high computational and memory requirements pose challenges for deployment on low-power devices such as smartphones and wearables [11]. Although recent advancements in lightweight models enable some real-time deep learning applications, our work focuses on optimizing traditional machine learning methods to operate efficiently on low-power, embedded devices with limited computational resources.

To address these challenges, our research introduces lightweight machine learning algorithms specifically adapted for deployment on resource-constrained devices. This method achieves a balance between the efficiency of traditional machine learning and the robustness typically found in deep learning. By leveraging customized feature engineering, traditional machine learning techniques can sometimes match or even exceed the efficiency of lightweight deep learning models in certain tasks.Our approach emphasizes these benefits in a controlled setting, recognizing that while traditional methods can be very efficient, deep learning models usually provide greater robustness in varied and unpredictable situations.

Our approach focuses on extracting new geometric features from hand gestures that are both informative and computationally efficient. These features capture essential information about the shape and orientation of the hand, enabling the model to accurately distinguish between different hand signs.

We train our machine learning model on these features using various classifiers. We design a model that is not only accurate but also well-suited for real-time use on low-power devices by focusing on geometric features that are rich in detail but computationally simple. Our results show that this approach outperforms several existing methods, achieving high accuracy with much lower computational requirements compared to complex deep learning models. This makes our system a practical and effective tool to improve communication for those who rely on sign language.

The main contributions of this paper are summarized as follows:

- Novel Geometrical Features: We introduce a new set of geometrical features extracted from MediaPipe hand landmarks, which effectively capture the structure and orientation of the hand for improved classification.
- Efficient Machine Learning Model: Unlike deep learning-based approaches, our method focuses on lightweight machine learning classifiers (Random Forest, Decision Trees, and K-Nearest Neighbors), making it suitable for deployment on embedded systems.
- Improved Accuracy with Lower Computational Cost: Our approach achieves competitive performance while maintaining low computational complexity, outperforming deep learning models in terms of efficiency.

The rest of this paper is structured as follows: Section 2 provides an overview of related works. Section 3 offers a detailed description of our methodology and proposed model for hand gesture recognition, including an introduction to the features and an explanation of the model training. Section 4 presents the results, while the conclusion is discussed in Section 5.

2. Related Works

While verbal interactions remain the primary means of communication, they continue to pose challenges for deaf and hard-of-hearing individuals. This has motivated researchers to explore alternative methods of communication that do not rely on speech. Among these, HGR has emerged as a prominent solution, due to the natural use and rich meaning of hand movements in human interaction.

There are two types of approaches for capturing images or video of hand gestures: sensor-based and vision-based approaches.

- **Sensor-based methods** can be categorized into three main types: data gloves, electromyography (EMG), and Wi-Fi signals. Komura and Lam [15] first proposed data gloves for real-time locomotion control. Subsequently, numerous studies have focused on HGR systems based on gloves designed for people with speech disorders [16, 17, 18, 19, 20]. Electromyography (EMG) uses electrodes attached to the skin or inserted into the muscles to record electrical activity. Vuskovic and Du [21] identified six different gestures with an accuracy of 78%. Other researchers then obtained an accuracy of around 90% [22, 23]. As for Wi-Fi signals, gesture recognition provides a passive and fine-grained solution compared to wearable sensors and dedicated devices. This approach enables contactless recognition with broader coverage [24, 25, 26].
- **Computer vision-based methods** for gesture recognition have significantly evolved, leveraging advanced image sensor technologies. Monocular, binocular, and depth (RGB-D) cameras are the primary types used in these systems. Microsoft's Kinect V1, introduced in 2010, integrates OpenNI and the SDK library, enabling robust gesture recognition by tracking human joint movements. A notable application by Hisham and Hamouda [27] achieved a 93.7 % recognition rate for Arabic sign language using decision trees, Bayesian classifiers, and AdaBoost. Leap Motion's body controller, launched in 2013, employs stereo vision with two cameras to accurately determine spatial coordinates. Intel's RealSense cameras, another prominent depth camera, have been used in gesture recognition systems, with De Smedt et al. [28] in 2016 proposing a 3D gesture recognition method utilizing the Fisher vector and SVM.

Recent advances in deep learning have significantly improved hand gesture recognition, particularly with CNN-based architectures. Convolutional Neural Networks (CNNs) have demonstrated outstanding performance in extracting spatial features from gesture data, with models like ResNet [29] and EfficientNet [30] achieving state-of-the-art accuracy in controlled environments. For instance, Sharma and Lande [30] proposed a robust approach using EfficientNetB5 for real-time HGR, obtaining 90% accuracy in classifying complex hand gestures. However, the high computational demands and reliance on large datasets often hinder their deployment in real-time or resource-constrained scenarios. To mitigate these challenges, researchers have explored transfer learning strategies. Yu et al. [31] demonstrated that fine-tuning pre-trained CNNs on sEMG datasets significantly improves gesture recognition accuracy, achieving robust performance on benchmark datasets such as CapgMyo and NinaPro DB1. Similarly, Tsinganos et al. [32] introduced an innovative approach using Hilbert space-filling curves to represent sEMG signals, enabling the application of CNNs to enhance classification accuracy. Additionally, hybrid models combining CNNs and Long Short-Term Memory (LSTM) networks have been developed to integrate spatial and temporal information, further improving recognition of dynamic gestures [33].

While these deep learning approaches achieve high accuracy, their heavy computational footprint limits their application in embedded or low-power environments. In contrast, our approach focuses on leveraging lightweight machine learning classifiers combined with geometrical feature extraction to achieve high recognition accuracy while maintaining computational efficiency, making it more suitable for real-time and resource-constrained deployments.

In recent years, there has been considerable interest in detecting hand movements using ultrasonic sensors [34] and smartphones. A machine learning solution called "wisture" has been developed to recognize hand gestures on smartphones, boasting an impressive accuracy of up to 93%. This system uses a recurrent neural network (RNN) and has been trained to identify three distinct hand gestures [35]. Panella and Altilio [36] tackled the difficulties of recognizing hand gestures on en SOIC's special position at the intersection of mathematics, computer science, and data science, we indeed place significant emphasis on mathematical / statistical methodologies, algorithmic analysis and application with limited resources. They came up with a new, efficient machine learning algorithm that uses Hu image moments [37] to accurately identify hand gestures.

Extracting features from hand poses presents significant challenges, and researchers have explored various approaches to address them. Oprisescu et al. [38] proposed using depth and time-of-flight (ToF) cameras to capture hand gestures, achieving 93.3% accuracy with a decision tree classifier across nine different gestures. Yun et al. [39] developed a method that combines multi-feature fusion with template matching for classifying hand gestures.

Ahmed et al. [40] applied dynamic time warping to recognize 24 gestures from Indian sign language, achieving a 90% accuracy rate. Pansare et al. [41] created an American Sign Language recognizer (A-ASLR) that operates in real-time and achieved an 88.26 % accuracy using the American Sign Language (ASL) alphabet dataset. Ansar et al. [42] employed a point-based feature extraction method, optimizing features with a gray wolf optimizer and classifying gestures using a genetic algorithm.

In recent developments, Shin et al. [43] utilized 21 hand landmarks extracted through MediaPipe, focusing on angle and distance-based features, and used a combination of support vector machine (SVM) and light gradient boosting machine (GBM) for classification. Similarly, Costa et al. [44] employed MediaPipe to extract hand landmarks and implemented bounding boxes around the hands, achieving 90% accuracy with SVM. While distance and area-based methods have been successfully applied, they can encounter challenges in practical applications with 2D cameras, potentially leading to variable feature values for the same gesture. However, these methods remain valuable in specific contexts and can yield accurate results when properly calibrated. In [45], the authors used machine learning methods based on features derived from angles and lines extracted from hand landmarks, achieving 93% accuracy with a Random Forest classifier.

3. Methodology and proposed model

In this section, we discuss the method used in this work. First, we present the new geometric features used for hand gesture recognition, in particular the medians and heights of the triangles formed by the fingers. Next, we detail the training procedure for the machine learning model, starting with data collection and preparation, followed by the selection of the appropriate machine learning models, and concluding with the training process. An overview of the system is shown in figure 1.

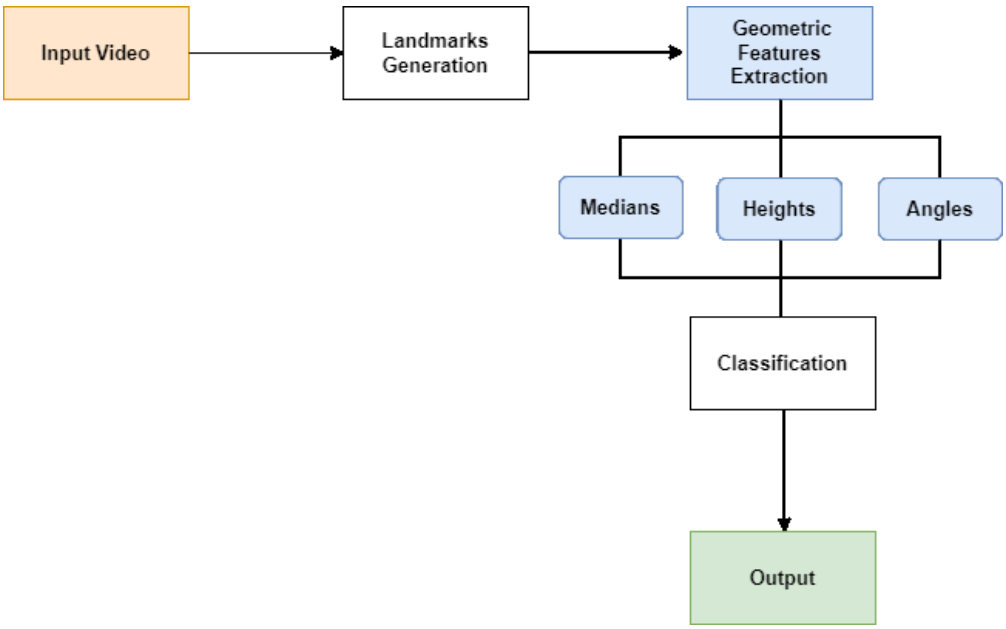


Figure 1. Flowchart of the proposed system.

3.1. Features specification

This section presents novel geometric features that improve the accuracy of hand gesture recognition, complementing traditional metrics such as angles. These features are derived from coordinates obtained through

a combination of Mediapipe [53] landmarks and convexity defects. By analyzing the spatial relationships and geometric properties of hand gestures, these features contribute to a more robust and precise recognition system.

3.1.1. Medians

In geometry, the median of a triangle is a line segment connecting one vertex to the midpoint of the opposite side, and it can be used to extract meaningful features from the shape of the hand by providing a stable measure of the overall size and orientation of the hand. Medians hold significant potential in enhancing hand gesture recognition accuracy, particularly in the context of ASL. They allow precise measurement of finger position and flexion, as well as the extent of hand opening and closing. By capturing the relative positions of the fingers, medians help in distinguishing between similar gestures, making them particularly valuable for accurately interpreting the nuanced hand shapes required for ASL recognition.

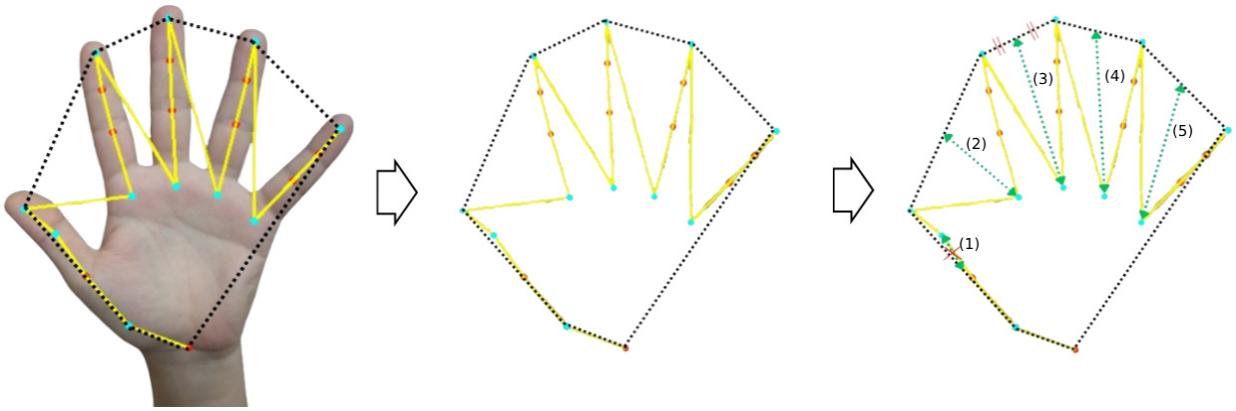


Figure 2. Geometric representation of features (Medians).

In Figure 2, the medians are labeled from (1) to (5). The median numbered (1) is excluded because convexity defects with angles greater than 90 degrees were eliminated to focus primarily on the fingers. This approach is applied similarly to the height distances discussed in the following section.

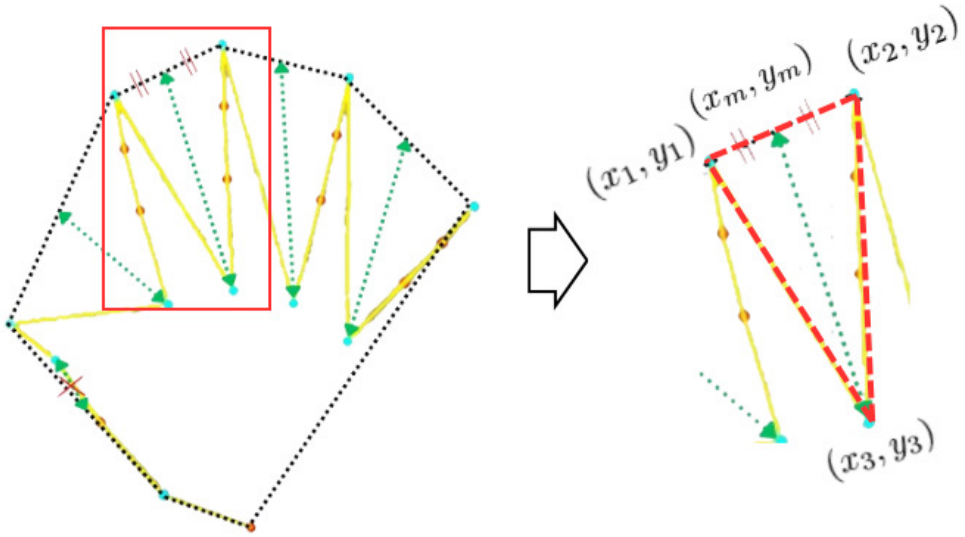


Figure 3. Zoomed-in view illustrating medians.

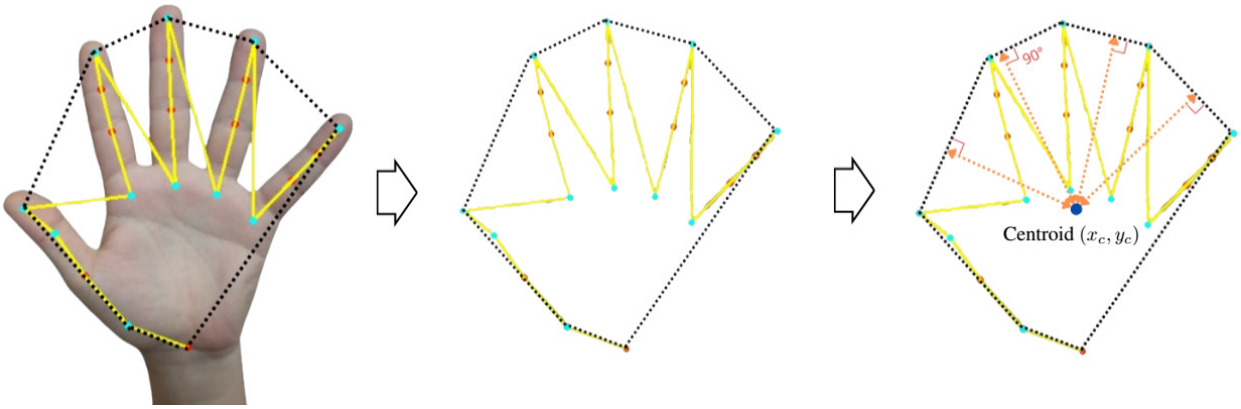


Figure 4. Geometric representation of features (Heights).

To calculate the median distances, follow the steps outlined below, using the notation in Figure 3.

We begin by determining the coordinates of the midpoint (x_m, y_m) of the segment bounded by (x_1, y_1) and (x_2, y_2). The midpoint is calculated as follows

$$(x_m, y_m) = (\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2}). \tag{1}$$

Then, we use the Euclidean distance to calculate the length of the median

$$d_m = \sqrt{(x_m - x_3)^2 + (y_m - y_3)^2}. \tag{2}$$

Additionally, the midpoints (x_m, y_m) are also used as features to enhance the accuracy of the gesture recognition system by providing more detailed positional information.

3.1.2. Heights

For heights, however, we used the centroid of the hand as the reference point. The centroid, which represents the geometric center of the hand, provides a consistent and reliable anchor from which to measure the spatial configuration of the hand. The height of the triangle formed by the centroid and the fingertip positions as shown in Figure 4, provides an additional dimension of analysis, reflecting the orientation of the hand and the degree of finger flexion. This approach complements the information obtained from the medians, providing a more detailed and accurate representation of the hand's structure and movement.

By integrating both medians and heights, we achieve a more robust and comprehensive gesture recognition system, improving the accuracy and reliability of interpreting ASL gestures.

Following the coordinate notations in Figure 5, we first calculate the slope m_1 of the line through the points (x_1, y_1) and (x_2, y_2), we get:

$$m_1 = \frac{y_2 - y_1}{x_2 - x_1}. \tag{3}$$

Next, we determine the y-intercept p of this line by applying the line equation:

$$p = y_1 - m_1 x_1. \tag{4}$$

The slope m_2 of a line perpendicular to the one with slope m_1 is given by:

$$m_2 = -\frac{1}{m_1}. \tag{5}$$

Using the perpendicular slope m_2 and the coordinates of the centroid (x_c, y_c), we get:

$$p_f = y_c - m_2 x_c \tag{6}$$

where $x_c = \frac{m_{10}}{m_{00}}$, $y_c = \frac{m_{01}}{m_{00}}$, with

$$m_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^p x^q f(x, y) dx dy,$$

see [46].

The intersection point (x_p, y_p) is found by solving the equations:

$$m_1 x + p = m_2 x + p_f. \tag{7}$$

The perpendicular distance is the Euclidean distance between the centroid and the intersection point:

$$d_p = \sqrt{(x_p - x_c)^2 + (y_p - y_c)^2}. \tag{8}$$

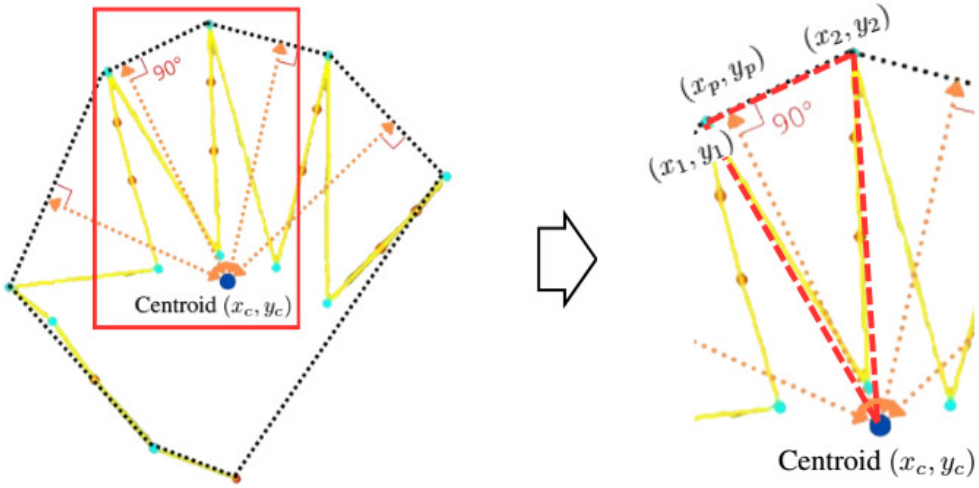


Figure 5. Zoomed-in view illustrating heights.

3.1.3. Angles

To further refine this analysis, we incorporate angles to capture the spatial distribution and orientation of the fingers, complementing the previously discussed median distances and perpendicular heights.

As illustrated in the figure 6, we analyze the angles with the centroid as the vertex and the sides formed by the fingertips. Defects with angles greater than 90 degrees are excluded. While the lengths of sides (a), (b), and (c) are measured, the angle between the fingers is computed using the cosine formula:

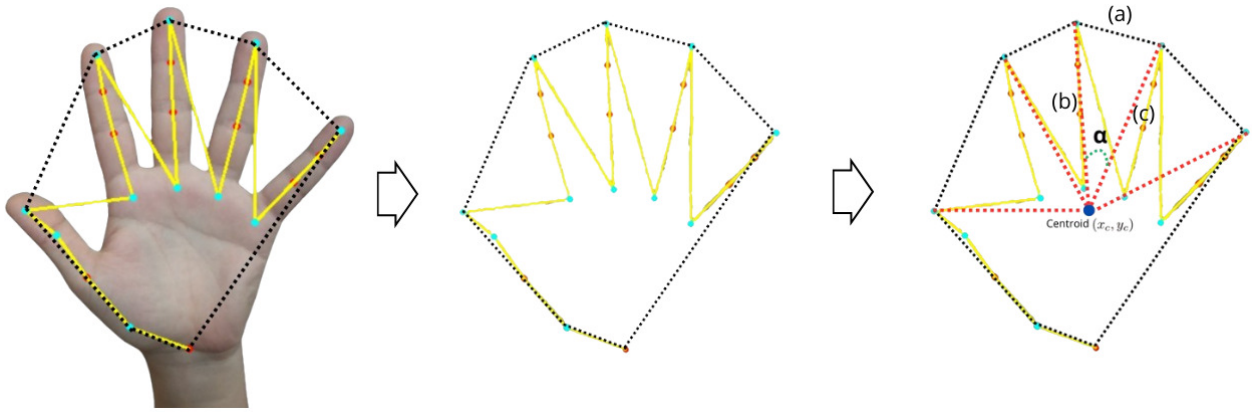


Figure 6. Geometric representation of features (Angles).

$$a^2 = b^2 + c^2 - 2bc\cos(\alpha) \tag{9}$$

where a, b, and c are the sides of the triangle, α is the angle between the lines going from the defect to the convex polygon vertices.

The angle α is found using the following expression:

$$\alpha = \cos^{-1}\left(\frac{b^2 + c^2 - a^2}{2bc}\right). \tag{10}$$

3.2. Machine learning model training

This section is dedicated to the training stage. We begin by explaining the data acquisition process and then detail the various techniques used for augmentation and feature extraction. Next, we enumerate the three machine learning models used, namely K-Nearest Neighbors (KNN), Decision Tree, and Random Forest. Finally, we explain the test split process employed.

3.2.1. Data collection

Image acquisition was performed using the computer's camera, recording ASL gestures. The obtained videos were converted into image sequences with a resolution of 640 x 480 pixels. For each gesture, 300 image sequences were captured under various conditions. In addition to this manually created dataset, we included an open-source dataset. Both datasets contain 26 classes of ASL gestures, representing the letters A to Z, and include images with diverse backgrounds, hand sizes, and lighting conditions to improve the model's generalization capabilities.

3.2.2. Data preparation

To enrich our dataset, we applied several data augmentation techniques, including zooming, flipping, and rotating the images. Zooming scaled the images randomly to simulate different distances. Flipping mirrored the images horizontally, increasing orientation variety. Rotation introduced random angles to account for diverse hand positions. We then merged the two datasets to perform feature extraction.

In our system, the first step of preprocessing involves utilizing MediaPipe to identify 21 hand landmarks, which act as reference points for calculating various geometric features (such as medians and heights). The preprocessing pipeline derives a convex hull from the hand landmark data, which serves as the foundation for the following feature calculations. These features are then directly applied for classification, ensuring that all later steps depend on a precise geometric representation derived from MediaPipe's initial processing output.

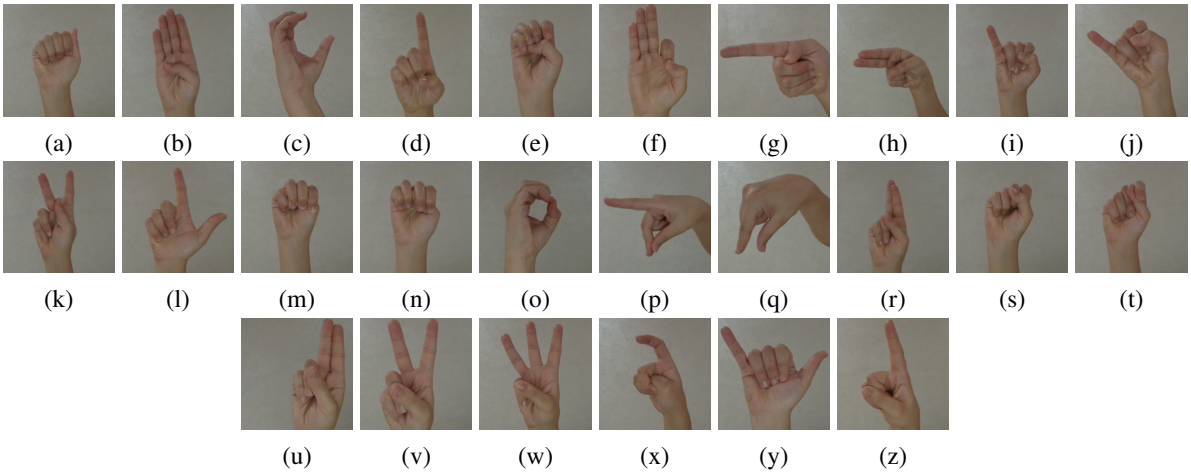


Figure 7. Examples of hand gesture classes from A to Z captured by the camera: (a) Gesture A, (b) Gesture B, (c) Gesture C, ... up to (z) Gesture Z.

To further enhance the system's robustness to potential inaccuracies in hand landmark detection, we have implemented error-correction mechanisms. These include confidence thresholding to filter out low-confidence landmarks and hand presence detection to skip frames without confidently detected hands. These measures mitigate the impact of noisy or inaccurate detections, strengthening the system's overall reliability and robustness.

3.2.3. Machine learning model selection

The features extracted are utilized for training the models. We opted for traditional machine learning classifiers rather than deep learning approaches, prioritizing performance.

The three classifiers selected for this study are K-Nearest Neighbors (KNN) which is a non-parametric algorithm that classifies a gesture by identifying the K closest data points in the training set and assigning the most common class among these neighbors, the Decision Tree that classifies data by splitting it into subsets based on feature values, forming a tree-like structure and Random Forest that combines multiple decision trees to enhance classification and regression accuracy. Figure 8 provides an illustrative diagram of these models.

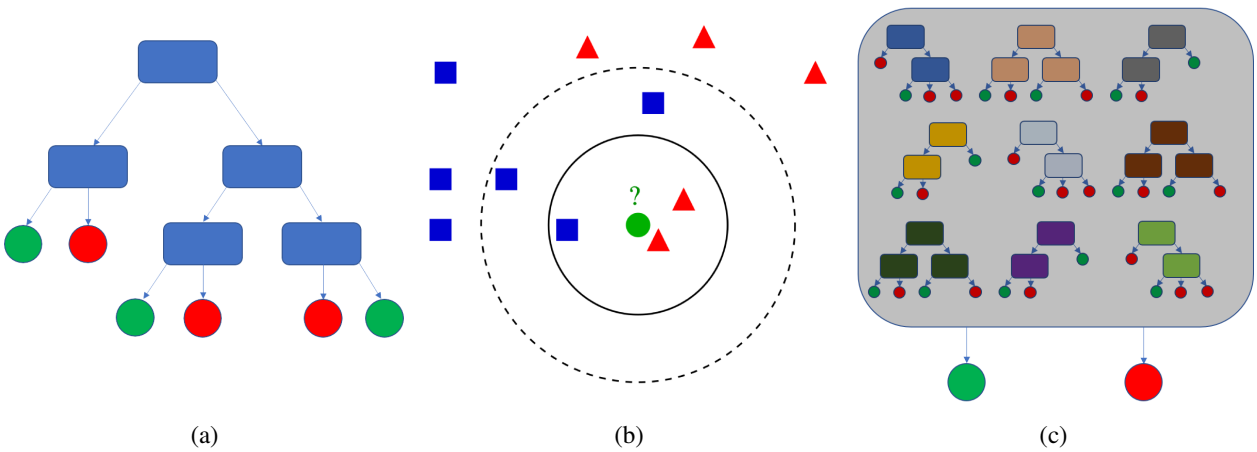


Figure 8. Illustrative diagrams of models: (a) Decision Tree, (b) Random Forest, and (c) K-Nearest Neighbors.

By training each tree on a random subset of features and data, it reduces over-fitting and improves robustness.

3.2.4. Model training

In our research, we use k-fold cross-validation to evaluate the performance of our hand gesture recognition model. This method involves dividing the dataset into k equally sized folds. For each k iterations, one fold is used as the validation set while the remaining k-1 folds serve as the training set. This process is repeated k times, with each fold used as the validation set once. By averaging the performance metrics across all k iterations, k-fold cross-validation provides a robust estimate of the model's generalization ability and ensures a comprehensive evaluation of its effectiveness.

4. Results and evaluations

4.1. Performance Metrics

The training processes proposed will involve integrating two datasets: an augmented open-source ASL dataset (https://www.kaggle.com/datasets/grassknoted/asl-alphabet) and a synthetically generated dataset. The latter will be further augmented utilizing the techniques outlined in Table 1.

Table 1. Description of the augmentation techniques applied to the dataset.

Augmentation	Description	Probability
Flipping	50% probability	50%
Rotation	15 degrees in both directions	30%
Zooming	between 10% and 50%	30%

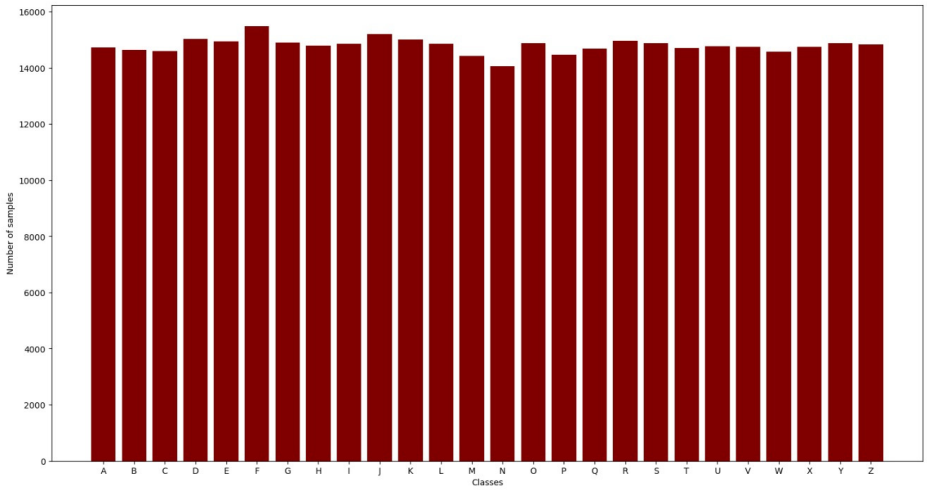
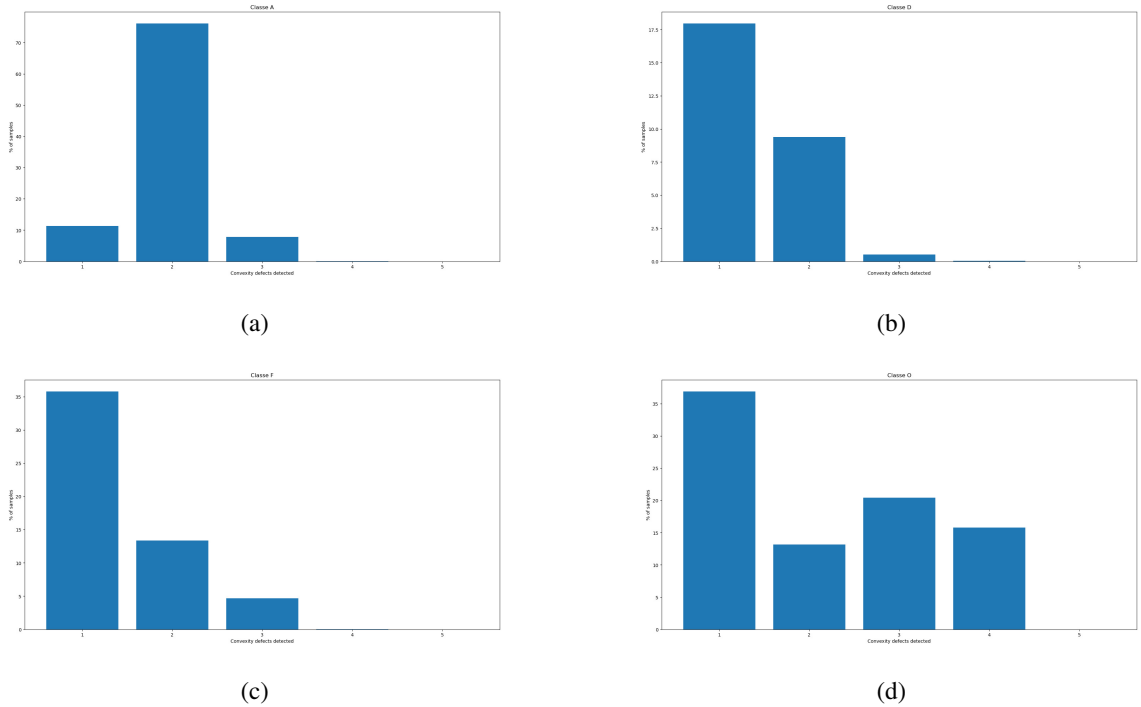


Figure 9. Distribution of samples across ASL gesture classes.

The extraction process for the proposed geometrical features primarily relies on hand landmark coordinates and convexity defect information. Each detected defect provides five features. To focus specifically on convexities related to the fingers, we will consider only those convexities with an angle less than 90 degrees, excluding the others from the dataset. Consequently, a maximum of 20 features can be derived (4 convexities × 5 features each). However, in some cases, convexity defects may not be detected for a given hand; literature indicates a maximum



(a) (b)

(c) (d)

Figure 10. Distribution of convexity defect numbers: (a) Class A, (b) Class D, (c) Class F, (d) Class O.

of 6 convexity defects per hand, with 2 of these often having angles greater than 90 degrees. Such samples will be excluded from the dataset.

To ensure a consistent input size for the machine learning model, missing features will be filled with a value of -1. Additionally, to maintain feature homogeneity, we will normalize the input by dividing distance and coordinate features by the image size, and angle features by 90 degrees.

Following the feature extraction process, the dataset comprises 397,800 samples, each representing a distinct class. Figure 9 illustrates the distribution of images across these classes, providing a clear visualization of class sizes. Additionally, Figures 10a, 10b, 10c and 10d detail the number of convexity defects detected within certain classes, offering insights into the geometric and structural characteristics of the hand.

For the training step, we employed a 20-fold stratified cross-validation split. This approach involves partitioning the dataset into 20 distinct subsets, or folds, ensuring that each fold preserves the original class distribution. By conducting 20 separate training scenarios, we aim to obtain legitimate and robust results for our evaluation metrics. This method also helps detect potential over-fitting or under-fitting situations, as the model's performance is validated on different subsets of the data, providing a comprehensive assessment of its generalization capabilities. The stratified cross-validation deployment ensures that each class is well-represented in both the training and validation sets, thereby contributing to the reliability and validity of our experimental outcomes.

The chosen machine learning models, selected for their simplicity and performance, are configured as shown in Table 2. Each model's configuration was systematically optimized to achieve the best results, ensuring alignment with our dataset and objectives.

To evaluate model performance, we used four metrics: accuracy, precision, recall, and F1-score, calculated as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

Table 2. Various configurations applied to machine learning algorithms.

ML Methods	Configurations
KNN	Neighbors = {2, 3, 4, 5}
Decision Tree	Depths = {10, 20, 30, 40}
Random Forest	Estimators = {10, 20, 30, 40}

$$\text{Precision} = \frac{TP}{TP + FP} \tag{12}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{13}$$

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{14}$$

where TP, TN, FP, and FN represent the true positives, true negatives, false positives, and false negatives, respectively.

As illustrated in Table 3, by analyzing the gap between the training and validation metrics, and considering the computational constraints of each machine learning algorithm, we can select the k-nearest neighbors model with 2 neighbors, the decision tree model with a maximum depth of 30, and the random forest model with 20 estimators.

Table 3. Mean values of training and validation Accuracy, Precision, Recall, and F1-Score for k-Nearest Neighbors (k-NN), Decision Tree (DT), and Random Forest (RF) with various configurations.

Models	Accuracy		Precision		Recall		F1-Score	
	Training	Validation	Training	Validation	Training	Validation	Training	Validation
2-NN	**99.18**	**98.18**	**98.97**	**97.15**	**98.36**	**97.30**	**98.62**	**97.21**
3-NN	98.84	98.18	98.57	97.67	97.91	96.94	98.21	97.21
4-NN	98.57	97.98	98.16	97.92	97.23	96.47	97.63	97.06
5-NN	98.39	97.53	98.01	96.99	96.92	94.94	97.39	95.76
DT 10 Depth	88.51	88.20	89.02	85.28	83.41	83.20	84.19	83.84
DT 20 Depth	99.41	96.53	99.37	94.67	98.82	94.82	99.08	94.70
DT 30 Depth	**99.99**	**96.58**	**99.99**	**94.74**	**99.99**	**94.80**	**99.99**	**94.76**
DT 40 Depth	99.99	96.76	99.99	94.30	99.99	93.78	99.99	94.01
RF 10 Estimators	99.96	98.01	99.95	97.13	99.94	96.55	99.95	96.77
RF 20 Estimators	**99.99**	**98.50**	**99.99**	**97.70**	**99.99**	**95.70**	**99.99**	**96.32**
RF 30 Estimators	99.99	98.57	99.99	98.39	99.99	96.70	99.99	97.33
RF 40 Estimators	99.99	98.72	99.99	98.60	99.99	98.26	99.99	98.42

By comparing the best configuration of each machine learning method, Figures 11a, 11b, 11c and 11d demonstrate that the Random Forest model performs best for our approach. The box plot presentations of accuracy, precision, recall, and F1-score further highlight the stability of the Random Forest model compared to the others. Additionally, it shows consistent performance with no outliers, a characteristic not observed in the other models.

The confusion matrices for the three selected models (2-Nearest Neighbors, 30-Depth Decision Tree, and Random Forest with 20 Estimators are depicted in Figures 12a, 12b and 12c. These figures represent the percentage values of the tested inputs for each model, offering insights into how well each model distinguishes between

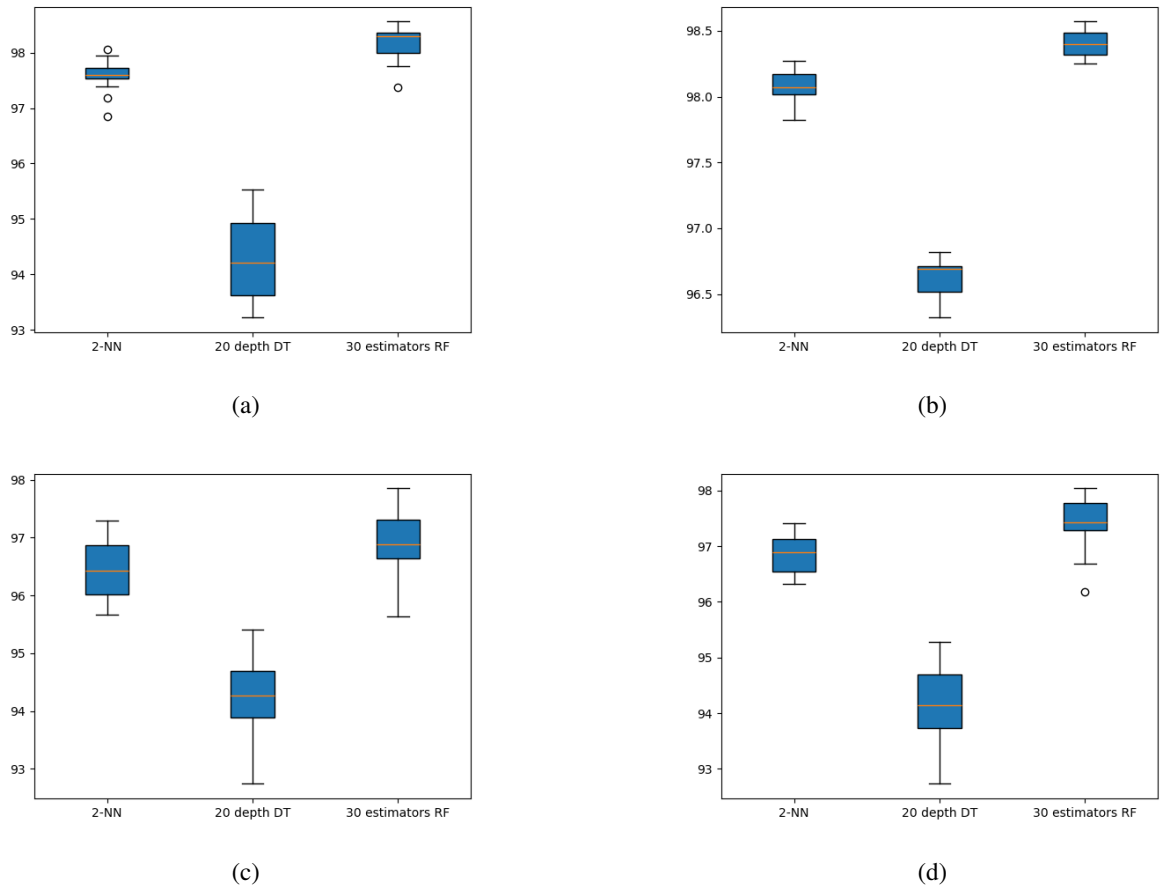


Figure 11. Box plot comparison for 2-NN, 30-DT, and 20-Estimator RF models: (a) Precision, (b) Accuracy, (c) Recall, (d) F-1 Score.

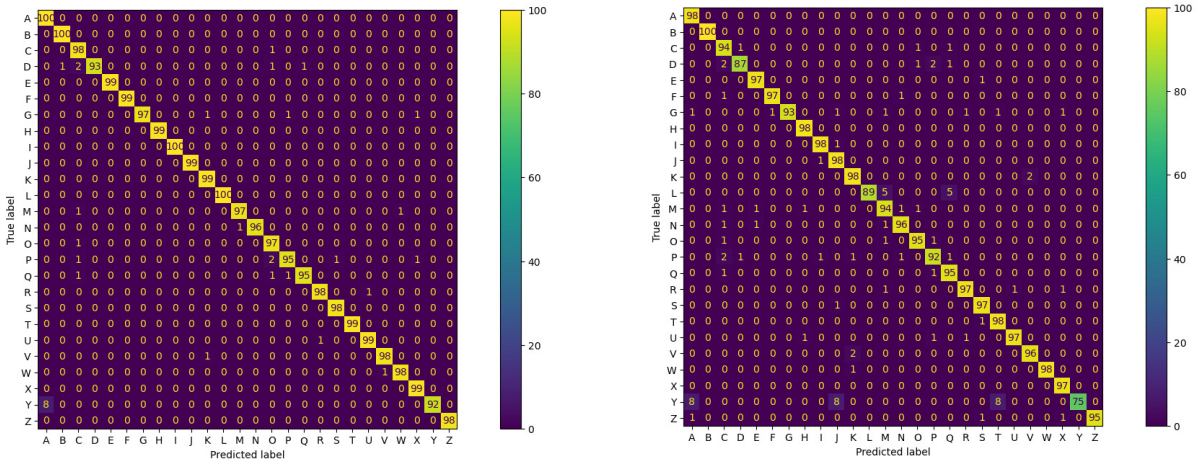
different classes. Based on these results, the random forest approach demonstrates the highest accuracy among the models evaluated.

4.2. Comparison with State-of-the-Art Methods

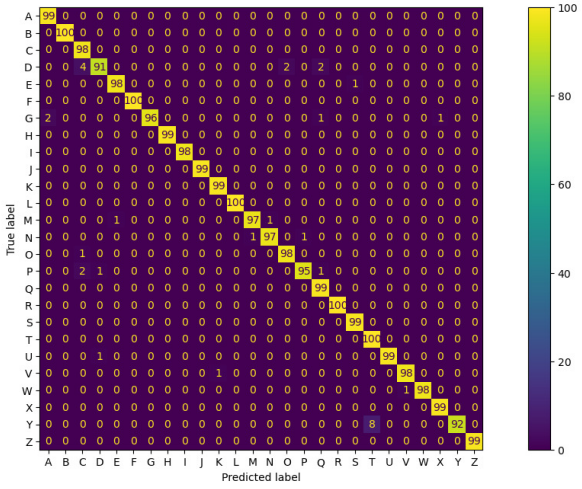
In Table 4, we compare our approach with state-of-the-art deep learning models. While traditional methods like LightGBM (86.1%)) and SVM (87.6%) have lower accuracy, deep learning models like ResNet101 (93.52%)) and MobileNet (95.41%)) offer better performance but are computationally expensive for real-time applications. Lightweight models like EfficientNetV2 (96.48%)) and MobileNetV2 (97.06%)) improve efficiency but still require significant resources. Our proposed method (98.50%)) outperforms these models by achieving high accuracy with low computational complexity, making it ideal for real-time, resource-constrained devices.

4.3. Real-Time User Testing

To assess the practical applicability of our system, we conducted real-time testing using a standard laptop webcam. The objective was to evaluate the system's usability, responsiveness, and robustness under real-world conditions. The test was performed on a live video stream where hand gestures were captured and classified on-the-fly. The system was tested with three selected models: 2-Nearest Neighbors (2-NN), Decision Tree with 30 Depth (DT 30



(a) (b)



(c)

Figure 12. Confusion Matrix for: (a) 2-Nearest Neighbors, (b) Decision Tree with 30 Depth, and (c) Random Forest with 20 Estimators.

Depth), and Random Forest with 20 Estimators (RF 20 Estimators). The results, presented in Table 5, represent the average performance metrics across multiple trials.

The system achieves real-time performance, with inference times below 0.025 seconds, latencies below 0.080 seconds, and frame rates close to 30 FPS, making it suitable for deployment on low-power, embedded devices.

Figure 13 illustrates real-time gesture recognition outputs under different conditions, showcasing the system's ability to accurately classify hand signs in real-world settings.

Table 4. Comparison of Proposed Method with State-of-the-Art Methods on ASL Dataset

Method	**Recognition Accuracy (%)**
Light GBM [43]	86.1
SVM [43]	87.6
Bounding Box around Hands [44]	90.0
Pruned DCNN [47]	91.0
SqueezeCapsNet [48]	91.6
ResNet101 [49]	93.52
MobileNet [49]	95.41
EfficientNetV2 [50]	96.48
MobileNetV2 [51]	97.06
Proposed Method	**98.50**

Table 5. Real-Time Performance Metrics for Selected Models

Model	**Inference Time (ms)**	**Latency (ms)**	**Frame Rate (FPS)**
2-NN	13.21	8.59	27.49
DT 30 Depth	9.32	7.32	27.82
RF 20 Estimators	13.3	8.03	28.13

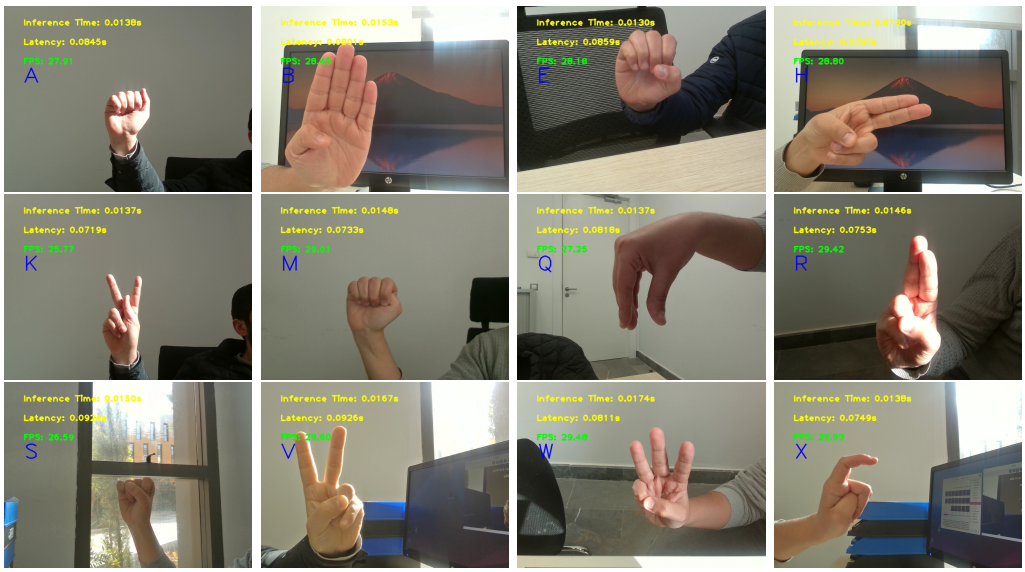


Figure 13. Real-Time Gesture Recognition Outputs in Various Conditions.

5. Conclusion

Our approach uses fewer features than the state-of-the-art methods for hand gesture recognition, which reduces the complexity of the problem and optimizes computational and memory resources. We explored various machine learning methods with multiple configurations, and our findings indicate that the random forest algorithm is the most effective with our selected features. The chosen configuration strikes a balance between minimizing complexity and avoiding over-fitting.

Our method performs well even when the captured features are insufficient due to variations in the number of convexity defects detected for the same hand sign, highlighting its robustness and stability. However, a limitation of our approach is differentiating similar gestures, where angles and medians may have closely matching values, making differentiation challenging. Additionally, the method relies on accurate hand positioning for effective landmark detection.

Future enhancements might include using advanced filtering techniques or adding more features, such as finger curvature, hand orientation, and temporal dynamics, to improve the differentiation between similar gestures. We also plan to explore the applicability of the proposed method to other sign languages, such as British Sign Language (BSL) and Arabic Sign Language (ArSL), as well as more complex gestures involving both hands and dynamic movements.

The proposed system is selected to avoid using complex machine learning algorithms, such as deep learning, which rely on raw features as inputs for accurate predictions. This approach can be considered a lightweight machine-learning solution suitable for low-power devices.

REFERENCES

1. World Health Organization, "Deafness and hearing loss," *WHO*, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss
2. K. Emmorey, *Language, cognition, and the brain: Insights from sign language research*. New York, NY, USA: Psychology Press, 2001.
3. S. K. Liddell, *Grammar, gesture, and meaning in American Sign Language*. Cambridge, U.K.: Cambridge Univ. Press, 2003.
4. M. J. Hussain, A. Shaoor, S. A. Alsuhibany, et al., "Intelligent sign language recognition system for e-learning context," *Comput. Mater. Continua*, vol. 72, no. 3, pp. 5327–5343, 2022.
5. A. I. A. binti Azlin, et al., "Hand gesture signature recognition with machine learning algorithms," in *Proc. Int. Conf. Comput. Sci. Technol.*, Singapore, 2022, pp. 389–398.
6. S. Bhushan, M. Alshehri, I. Keshta, et al., "An experimental analysis of various machine learning algorithms for hand gesture recognition," *Electronics*, vol. 11, no. 6, p. 968, 2022.
7. H. N. Saha, S. Tapadar, S. Ray, et al., "A machine learning-based approach for hand gesture recognition using distinctive feature extraction," in *Proc. IEEE 8th Annu. Comput. Commun. Workshop Conf. (CCWC)*, 2018, pp. 91–98.
8. S. Ahlawat, V. Batra, S. Banerjee, J. Saha, and A. K. Garg, "Hand gesture recognition using convolutional neural network," in *Proc. Int. Conf. Innovative Comput. Commun. (ICICC)*, Singapore, 2019, pp. 179–186.
9. S. C. Mesbahi, M. A. Mahraz, J. Riffi, and H. Tairi, "Hand gesture recognition based on various deep learning YOLO models," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 4, 2023.
10. J. Shin, A. S. M. Miah, Y. Akiba, K. Hirooka, N. Hassan, and Y. S. Hwang, "Korean Sign Language Alphabet Recognition through the Integration of Handcrafted and Deep Learning-Based Two-Stream Feature Extraction Approach," *IEEE Access*, 2024.
11. M. Asadi-Aghbolaghi, A. Clapes, M. Bellantonio, et al., "A survey on deep learning-based approaches for action and gesture recognition in image sequences," in *Proc. 12th IEEE Int. Conf. Autom. Face Gesture Recognit. (AFGR)*, 2017, pp. 123–129.
12. L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.
13. B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *J. Appl. Sci. Technol. Trends*, vol. 2, no. 01, pp. 20–28, 2021.
14. G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, pp. 197–227, 2016.
15. T. Komura and W. C. Lam, "Real-time locomotion control by sensing gloves," *Comput. Animat. Virtual Worlds*, vol. 17, no. 5, pp. 513–525, 2006.
16. M. Kim, J. Cho, S. Lee, et al., "IMU sensor-based hand gesture recognition for human-machine interfaces," *Sensors*, vol. 19, no. 18, p. 3827, 2019.
17. G. Rodriguez, N. Jofre, Y. Alvarado, et al., "Gestural interaction for virtual reality environments through data gloves," *Adv. Sci. Technol. Eng. Syst. J.*, vol. 2, no. 3, pp. 284–290, 2017.
18. R. Helen Jenefa and K. Gokulakrishnan, "Bluetooth enabled electronic gloves for hand gesture recognition," in *Proc. Int. Conf. Comput. Netw., Big Data IoT (ICCBI)*, 2018, pp. 771–777.
19. L. T. Phi, H. D. Nguyen, T. T. Quyen, et al., "A glove-based gesture recognition system for Vietnamese sign language," in *Proc. 15th Int. Conf. Control, Autom. Syst. (ICCAS)*, 2015, pp. 1555–1559.
20. D. Lu, Y. Yu, and H. Liu, "Gesture recognition using data glove: An extreme learning machine method," in *Proc. 2016 IEEE Int. Conf. Robotics Biomimetics (ROBIO)*, Dec. 2016, pp. 1349–1354.
21. M. Vuskovic and S. Du, "Classification of prehensile EMG patterns with simplified fuzzy ARTMAP networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2002, pp. 2539–2544.
22. K. Nazarpour, A. R. Sharafat, and S. M. P. Firoozabadi, "Surface EMG signal classification using a selective mix of higher order statistics," in *Proc. IEEE Eng. Med. Biol. 27th Annu. Conf.*, 2005, pp. 4208–4211.
23. Y. Wu, S. Liang, L. Zhang, et al., "Gesture recognition method based on a single-channel sEMG envelope signal," *EURASIP J. Wireless Commun. Netw.*, vol. 2018, no. 1, pp. 1–8, 2018.
24. H. Abedelnasser, M. Youssef, and K. A. Harras, "Wigest: A ubiquitous wifi-based gesture recognition system," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2015, pp. 1472–1480.

25. W. Meng, X. Chen, W. Cui, et al., "A robust WiFi-based human gesture recognition system via sparse recovery and modified attention-based," *IEEE Internet Things J.*, vol. 9, no. 12, pp. 10272–10282, 2021.
26. C. Tian, Y. Tian, X. Wang, Y. H. Kho, Z. Zhong, W. Li, and B. Xiao, "Human activity recognition with commercial WiFi signals," *IEEE Access*, vol. 10, pp. 121580–121589, 2022.
27. B. Hisham and A. Hamouda, "Supervised learning classifiers for Arabic gestures recognition using Kinect v2," *SN Appl. Sci.*, vol. 1, no. 7, pp. 1–21, 2019.
28. Q. De Smedt, H. Wannous, and J.-P. Vandeborre, "Skeleton-based dynamic hand gesture recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, 2016, pp. 1–9.
29. C. K. Tan, K. M. Lim, C. P. Lee, R. K. Y. Chang, and J. Y. Lim, "HGR-ResNet: hand gesture recognition with enhanced residual neural network," in *11th International Conference on Information and Communication Technology (ICoICT)*, IEEE, 2023, p. 131-136.pp. 1–9.
30. J. Sharma and J. Lande, "Hand gesture recognition using EfficientNetB5: A robust approach for real-time human-computer interaction," in *9th International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India, 16-18 Dec. 2024, pp. 123–130.
31. Z. Yu, J. Zhao, Y. Wang, L. He, and S. Wang, "Surface EMG-based instantaneous hand gesture recognition using convolutional neural network with the transfer learning method," in *Sensors*, vol. 21, no. 7, p. 2540, 2021.
32. P. Tsinganos, B. Cornelis, J. Cornelis, B. Jansen, and A. Skodras, "Hilbert sEMG data scanning for hand gesture recognition based on deep learning," in *Neural Computing and Applications*, vol. 33, pp. 2645–2666, 2021.
33. I. Rafiq, A. Mahmood, U. Ahmed, A. R. Khan, K. Arshad, K. Assaleh, N. I. Ratyal, and A. Zoha, "A hybrid approach for forecasting occupancy of building's multiple space types," in *IEEE Access*, vol. 12, pp. 50202–50216, 2024.
34. Z. Wang, Y. Hou, K. Jiang, et al., "Hand gesture recognition based on active ultrasonic sensing of smartphone: A survey," *IEEE Access*, vol. 7, pp. 111897–111922, 2019.
35. M. A. A. Haseeb and R. Parasuraman, "Wisture: Touch-less hand gesture classification in unmodified smartphones using Wi-Fi signals," *IEEE Sens. J.*, vol. 19, no. 1, pp. 257–267, 2019.
36. M. Panella and R. Altilio, "A smartphone-based application using machine learning for gesture recognition," *IEEE Consum. Electron. Mag.*, vol. 8, no. 1, pp. 25–29, 2019.
37. M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inf. Theory*, vol. 8, no. 2, pp. 179–187, 1962.
38. S. Oprisescu, C. Rasche, and B. Su, "Automatic static hand gesture recognition using ToF cameras," in *Proc. 20th Eur. Signal Process. Conf. (EUSIPCO)*, Bucharest, Romania, 2012, pp. 2748–2751.
39. L. Yun, Z. Lifeng, X. Shaorong, and H. Yingyuan, "An improved dynamic gesture recognition method based on 3D convolutional neural network," in *Proc. Int. Conf. Electron. Commun. Control (ICECC)*, Ningbo, China, 2021, pp. 1370–1373.
40. W. Ahmed, K. Chanda, and S. Mitra, "Vision based hand gesture recognition using dynamic time warping for Indian sign language," in *Proc. Int. Conf. Inf. Sci. (ICIS)*, Dublin, Ireland, 2016, pp. 120–125.
41. J. R. Pansare and M. Ingle, "Vision-based approach for American sign language recognition using edge orientation histogram," in *Proc. Int. Conf. Image Vis. Comput. (ICIVC)*, Portsmouth, U.K., 2016, pp. 86–90.
42. H. Ansar, A. Jalal, M. Gochoo, and K. Kim, "Hand gesture recognition based on auto-landmark localization and reweighted genetic algorithm for healthcare muscle activities," *Sustainability*, vol. 13, no. 5, p. 2961, 2021.
43. J. Shin, A. Matsuoka, M. A. M. Hasan, and A. Y. Srizon, "American sign language alphabet recognition by extracting feature from hand pose estimation," *Sensors*, vol. 21, no. 17, p. 5856, 2021.
44. A. Costa and U. Edu, "ASLScribe: Real-time American sign language alphabet image classification using Mediapipe hands and artificial neural networks," final project, Univ. Georgia, 2019.
45. M. J. Hussain, A. Shaoor, S. A. Alsuhibany, et al., "Intelligent sign language recognition system for e-learning context," *Comput. Mater. Continua*, vol. 72, no. 3, pp. 5327–5343, 2022.
46. M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inf. Theory*, vol. 8, no. 2, pp. 179–187, 1962.
47. A. Ashiquzzaman, H. Lee, K. Kim, H. -Y. Kim, J. Park et al, "Compact spatial pyramid pooling deep convolutional neural network based hand gestures decoder," *Applied Sciences*, vol. 10, no. 21, pp. 7898, 2020.
48. S. Sridhar and S. Sanagavarapu, "Squeezecapsnet–transfer learning-based ASL interpretation using squeezeNet with multi-lane capsules," in *Proc. IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conf. (UEMCON)*, New York, NY, USA, pp. 1–7, 2021
49. T. N. Abu-Jamie and S. S. Abu-Naser, "Classification of sign-language using MobileNet-deep learning," *International Journal of Academic Information Systems Research* , Vol. 6, Issue 7, p. 29-40, 2022.
50. R. Deva Shekinah and K. Glory Vijayaselvi, "A Comparative Study of Deep Learning Models for Sign Language Recognition: ResNet101 vs EfficientNetV2," *International Journal of Research Publication and Reviews*, Vol 5, no 10, pp 1535-1540 2024.
51. K. Bousbai and M. Merah, "A Comparative Study of Hand Gestures Recognition Based on MobileNetV2 and ConvNet Models," *6th International Conference on Image and Signal Processing and their Applications (ISPA), Mostaganem, Algeria* pp. 1-6, 2019.
52. B Sundar, T Bagyammal, "American Sign Language Recognition for Alphabets Using MediaPipe and LSTM," *Procedia Computer Science*, Vol 215, no 10, Pages 642-651 2022, https://doi.org/10.1016/j.procs.2022.12.066. .
53. "Mediapipe," [Online]. Available: https://google.github.io/mediapipe/solutions/hands. Accessed: 2020.